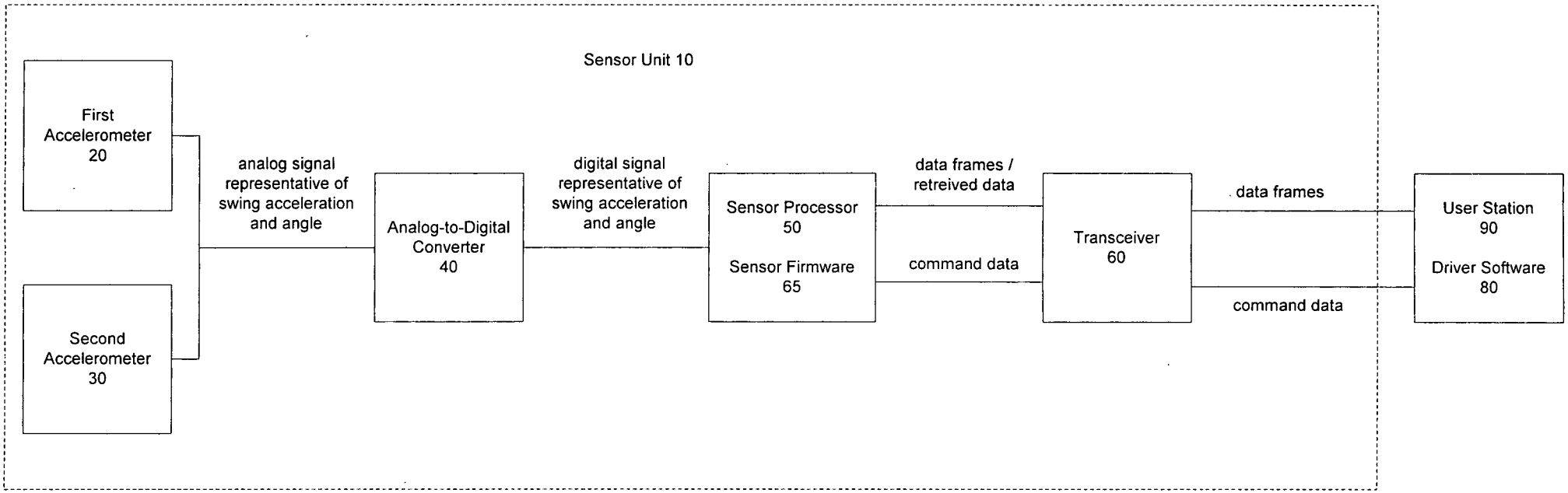
Fig. 1


Sensor Unit 10
First Accelerometer 20
Second Accelerometer 30
analog signal representative of swing acceleration and angle
Analog-to-Digital Converter 40
digital signal representative of swing acceleration and angle
Sensor Processor 50
Sensor Firmware 65
data frames / retreived data
command data
Transceiver 60
data frames
command data
User Station 90
Driver Software 80

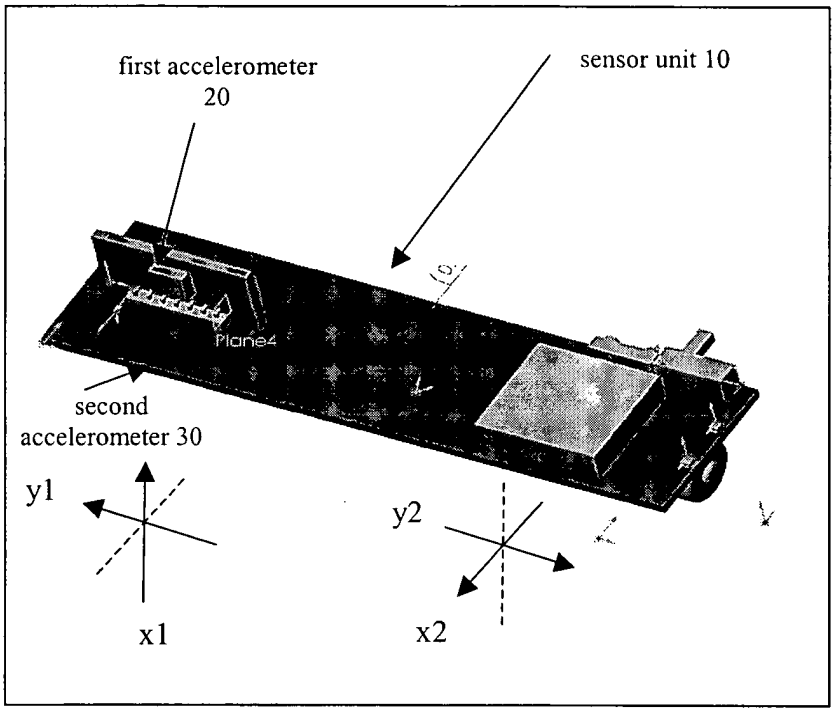

first accelerometer
20
sensor unit 10
Plane4
second
accelerometer 30
y1
x1
y2
x2

Fig. 2

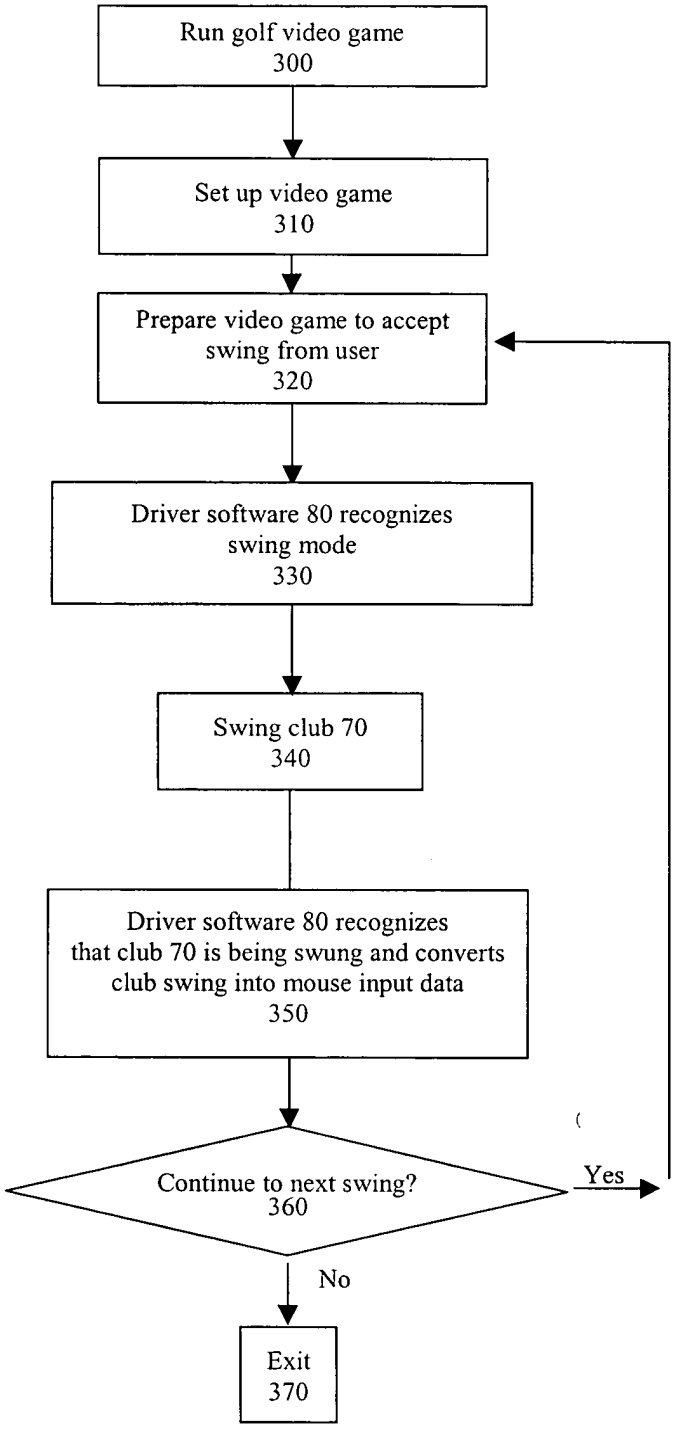

Run golf video game
300
Set up video game
310
Prepare video game to accept swing from user
320
Driver software 80 recognizes swing mode
330
Swing club 70
340
Driver software 80 recognizes that club 70 is being swung and converts club swing into mouse input data
350
Continue to next swing?
360
Yes
No
Exit
370

Fig. 3

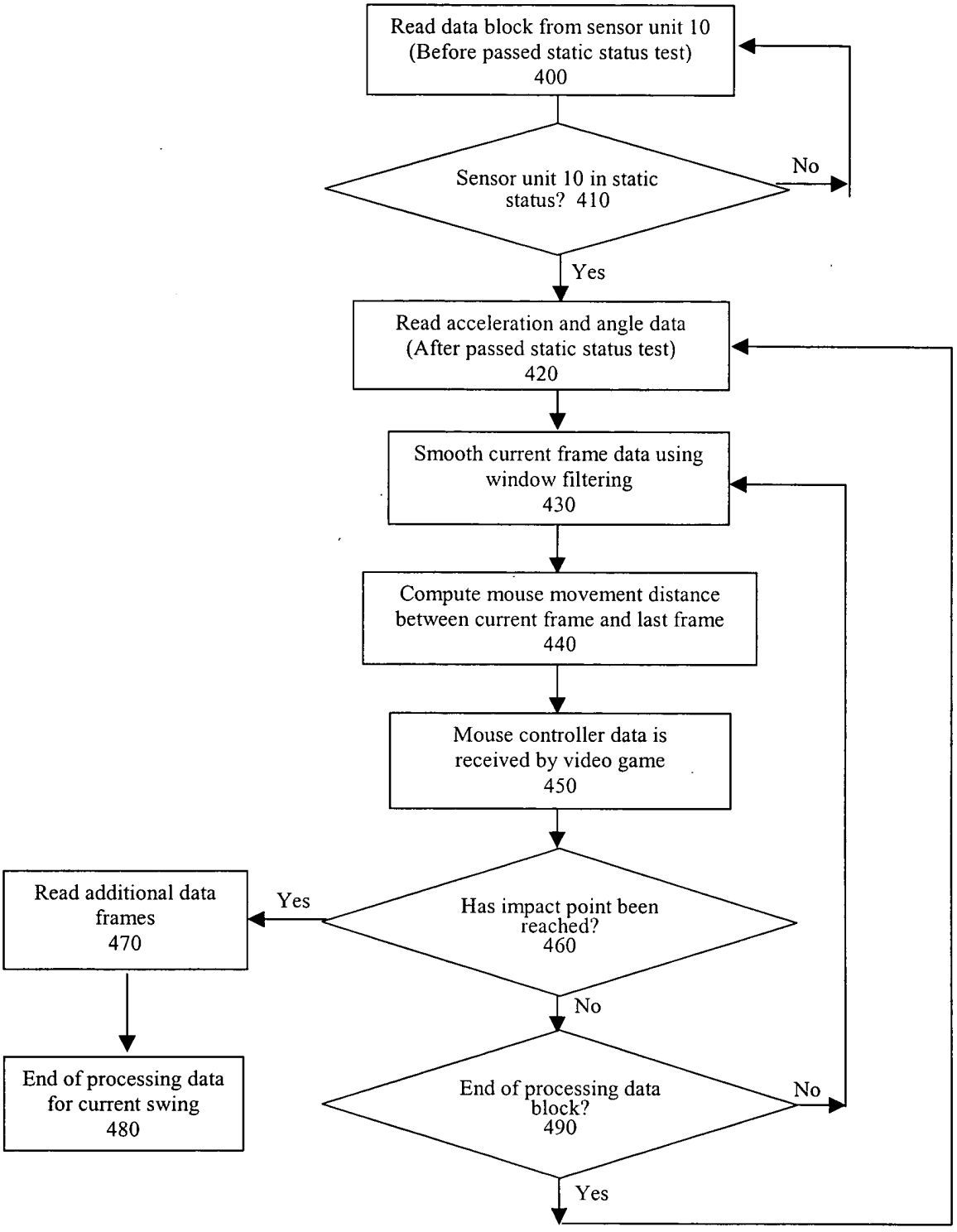

Read data block from sensor unit 10
(Before passed static status test)
400
Sensor unit 10 in static status? 410
No
Yes
Read acceleration and angle data
(After passed static status test)
420
Smooth current frame data using window filtering
430
Compute mouse movement distance between current frame and last frame
440
Mouse controller data is received by video game
450
Has impact point been reached?
460
Yes
Read additional data frames
470
No
End of processing data block?
490
No
Yes
End of processing data for current swing
480

Fig. 4

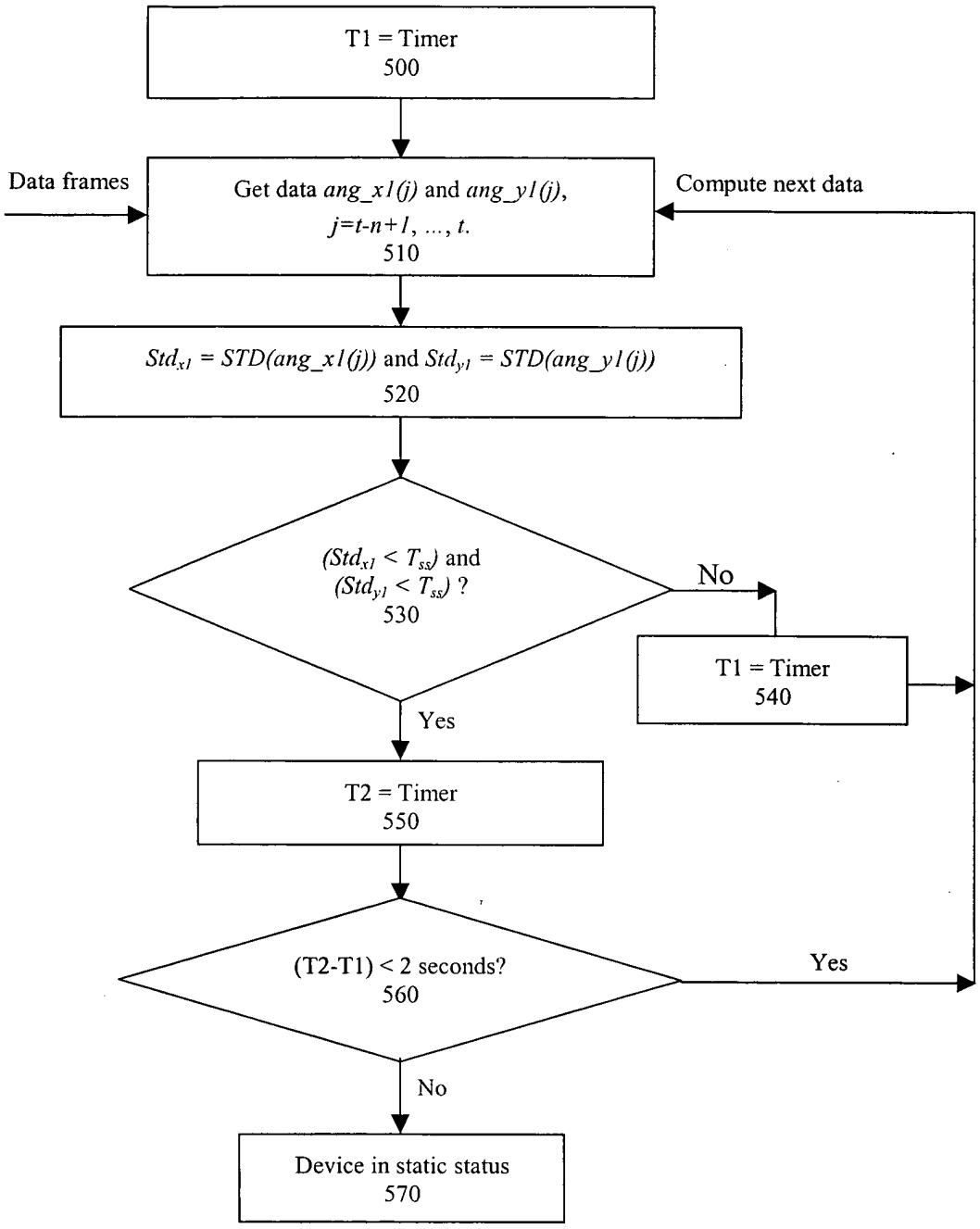

T1 = Timer
500
Data frames
Get data ang_x1(j) and ang_y1(j),
j=t-n+1, ..., t.
510
Compute next data
Std_x1 = STD(ang_x1(j)) and Std_y1 = STD(ang_y1(j))
520
(Std_x1 < T_ss) and
(Std_y1 < T_ss) ?
530
No
T1 = Timer
540
Yes
T2 = Timer
550
(T2-T1) < 2 seconds?
560
Yes
No
Device in static status
570

Fig. 5

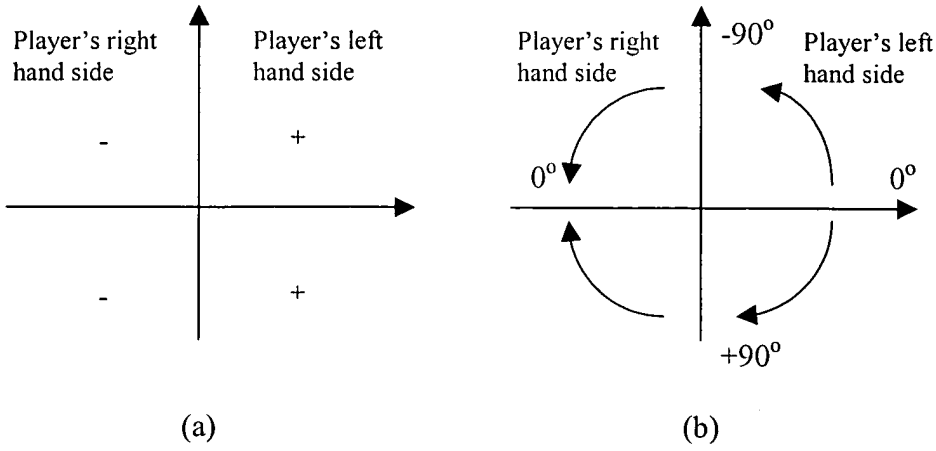

Player's right hand side
Player's left hand side
-
+
-
+
(a)
Player's right hand side
-90°
Player's left hand side
0°
0°
+90°
(b)

Fig. 6

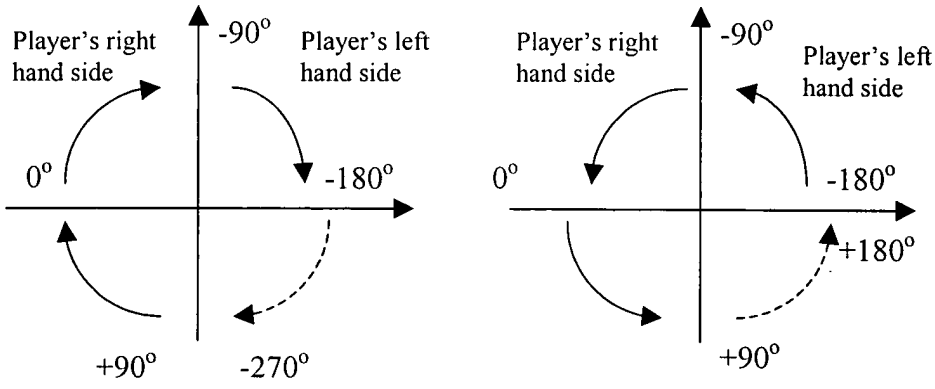

Player's right hand side
-90°
Player's left hand side
0°
-180°
+90°
-270°
(c)
Player's right hand side
-90°
Player's left hand side
0°
-180°
+180°
+90°
(d)

Fig. 6

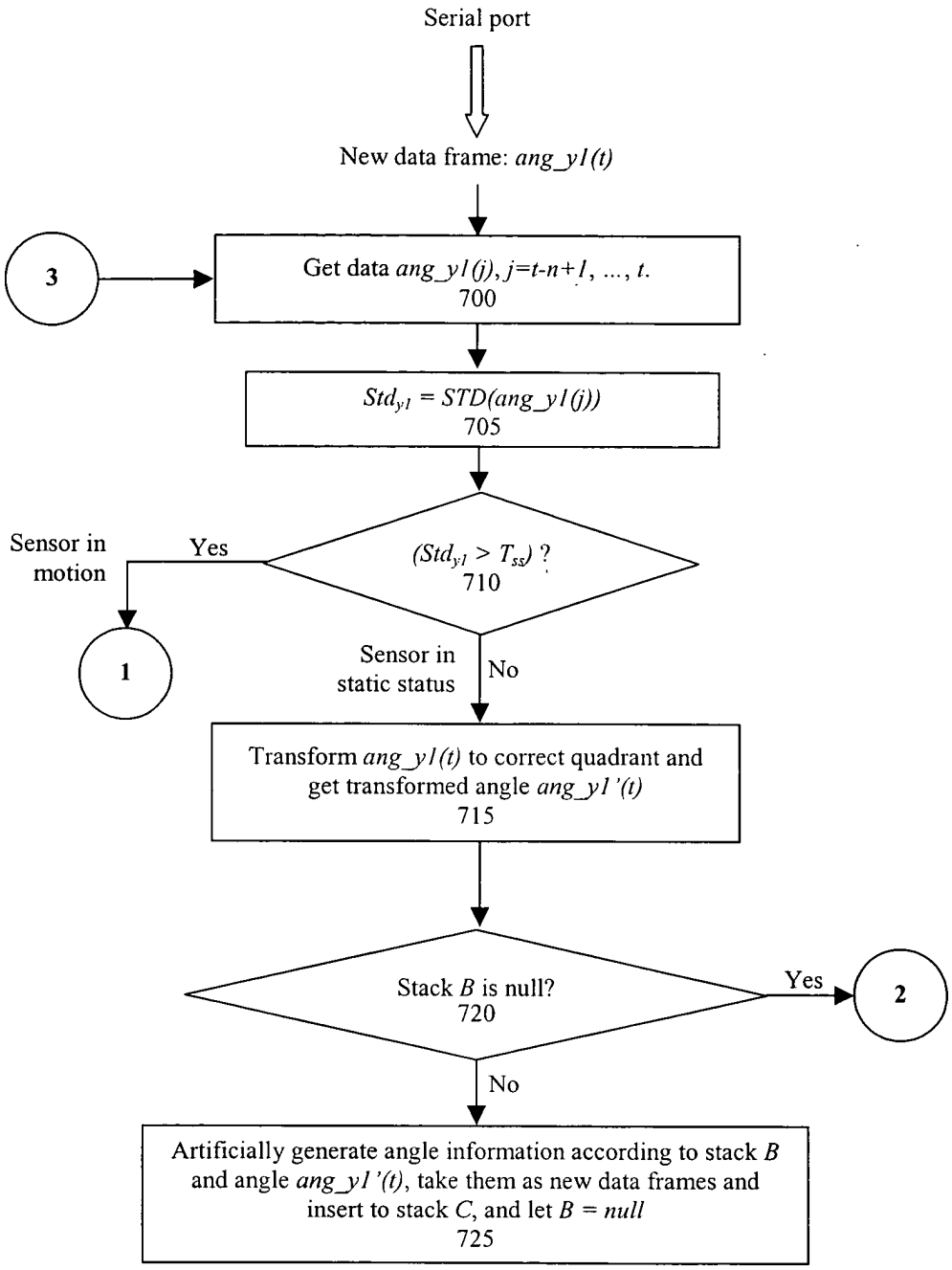

Serial port
New data frame: ang_y1(t)
3
Get data ang_y1(j), j=t-n+1, ..., t.
700
Std_y1 = STD(ang_y1(j))
705
Sensor in motion
Yes
(Std_y1 > T_ss) ?
710
1
Sensor in static status
No
Transform ang_y1(t) to correct quadrant and get transformed angle ang_y1'(t)
715
Stack B is null?
720
Yes
2
No
Artificially generate angle information according to stack B and angle ang_y1'(t), take them as new data frames and insert to stack C, and let B = null
725

Fig. 7(a)

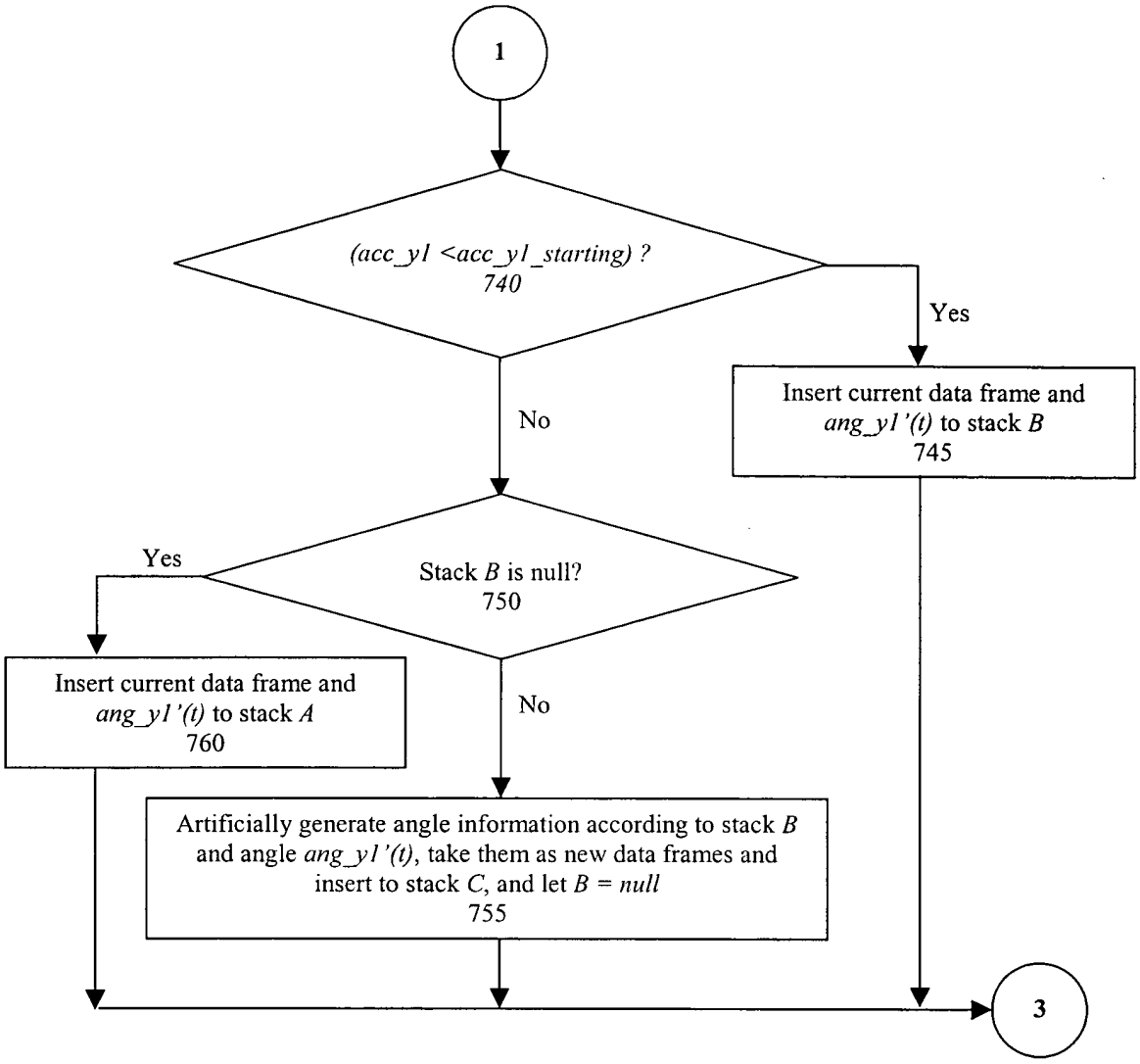

1
(acc_y1 <acc_y1_starting) ?
740
Yes
Insert current data frame and
ang_y1'(t) to stack B
745
No
Yes
Stack B is null?
750
Insert current data frame and
ang_y1'(t) to stack A
760
No
Artificially generate angle information according to stack B
and angle ang_y1'(t), take them as new data frames and
insert to stack C, and let B = null
755
3

Fig. 7(b)

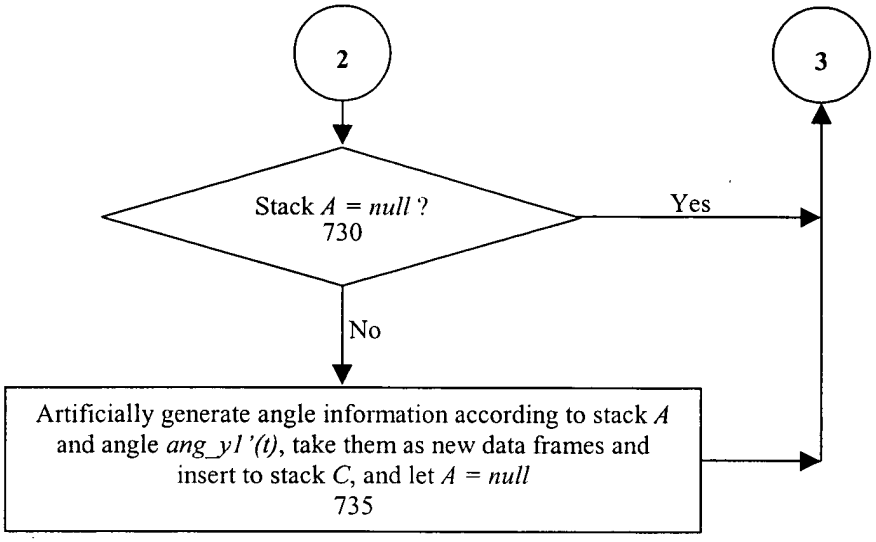

2
3
Stack A = null ?
730
Yes
No
Artificially generate angle information according to stack A and angle ang_y1'(t), take them as new data frames and insert to stack C, and let A = null
735

Fig. 7(c)

```
If ( ang_y1(t) >= 0 and club is swing up ) Then
        If ( ang_x2(t) > 0 ) Then ang_y1'(t) = – 180 – ang_y1(t)
ElseIf ( ang_y1(t) > ang_y1_starting – 60 and club is swing down and ang_x1(t)
<= 0 ) Then
        ang_y1'(t) = 180 – ang_y1(t)
ElseIf ( ang_y1(t) <= 0 and club is swing up ) Then
        If ( ang_x2(t) >= 0 ) Then
                ang_y1'(t) = – 180 – ang_y1(t)
        ElseIf ( ang_x2(t) < 0 ) Then
                ang_y1'(t) = ang_y1(t)
        End If
End If
```

Fig. 8

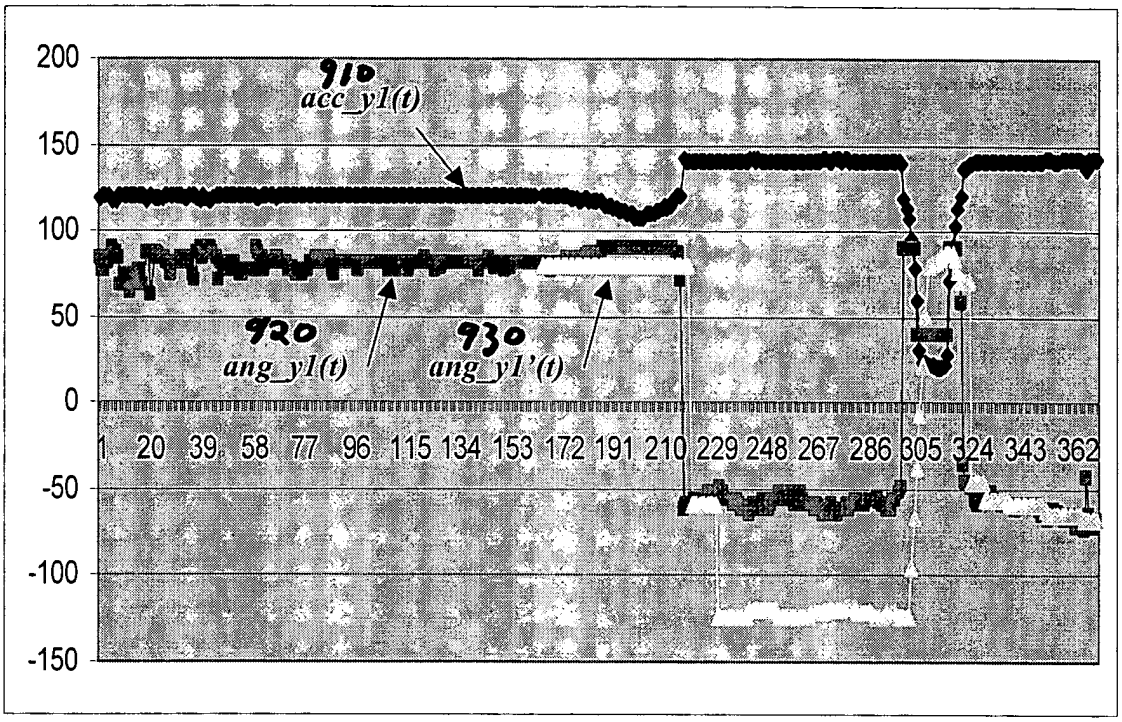

910
acc_y1(t)
920
ang_y1(t)
930
ang_y1'(t)
200
150
100
50
0
-50
-100
-150
1 20 39 58 77 96 115 134 153 172 191 210 229 248 267 286 305 324 343 362

Fig. 9

Input: angle_change and current_angle; Output: distance

1) **Let** distance = angle_change
2) **If** (swing down **And** current_angle > 90) **Then** distance = distance * 2
3) **If** (swing up) **Then**
4) Suppose a) current_angle < 45 **And** current_angle >= -45; or b) current_angle < -45 **And** current_angle >= -90; or c) current_angle < -90 **And** current_angle >= -145; or d) current_angle < -145 **And** current_angle >= -180; or e) current_angle < -180. Then **Let** R = 1.25, 1.5, 5, 7, 10 corresponding to a)-e) respectively.
5) **Let** distance = distance * R
6) **End If**
7) **If** (swing down) **Then**
8) Suppose a) current_angle <= -180; or b) current_angle <= -135 **And** current_angle > -180; or c) current_angle <= -90 **And** current_angle > -135; or d) current_angle > -90 **And** current_angle <= 0; or e) current_angle > 0 **And** current_angle <= 30; or f) current_angle > 30 **And** current_angle <= 90. Then **Let** R = 12, 10, 8, 6, 5, 5 corresponding to a)-f) respectively.
9) **Let** distance = distance / R
10) adjust distance value according to acceleration acc_y1.
11) **If** (distance value is small) **Then** adjust it according to the club's position
12) **End If**
13) **If** (club is not in motion) **Then Let** distance = 0
14) **If** (club passed starting position **And** distance < 5) **Then Let** distance = 5

Fig. 10(a)

Input: angle_change and current_angle; Output: distance

1) distance = angle_change
2) Suppose a) current_angle > starting_angle – 15; or b) current_angle > starting_angle - 30 **And** current_angle <= starting_angle – 15; or c) current_angle > starting_angle - 45 **And** current_angle <= starting_angle – 30; or d) current_angle > starting_angle - 60 **And** current_angle <= starting_angle – 45; or e) otherwise. **Then Let** R = 12, 12, 8, 8, 4 corresponding to a)-e) respectively.
3) **Let** distance = distance * R
4) **If** (swing down) **Then**
5) adjust distance value according to acceleration acc_y1.
6) **If** (distance value is small) **Then** adjust it according to the club's position
7) **End If**
8) **If** (club is not in motion) **Then Let** distance = 0
9) **If** (club passed starting position **And** distance < 5) **Then Let** distance = 5

Fig. 10(b)

Input: angle_change and current_angle; Output: distance

1) distance = angle_change
2) Suppose a) current_angle > starting_angle – 15; or b) current_angle > starting_angle - 30 **And** current_angle <= starting_angle – 15; or c) current_angle > starting_angle - 45 **And** current_angle <= starting_angle – 30; or d) current_angle > starting_angle - 60 **And** current_angle <= starting_angle – 45; or e) otherwise. **Then Let** R = 24, 24, 16, 16, 8 corresponding to a)-e) respectively.
3) **Let** distance = distance * R
4) **If** (swing down) **Then**
5) adjust distance value according to acceleration acc_y1.
6) **If** (distance value is small) **Then** adjust it according to the club's position
7) **End If**
8) **If** (club is not in motion) **Then Let** distance = 0
9) **If** (club passed starting position **And** distance < 5) **Then Let** distance = 5

Fig. 10(c)

```
Input: distance; Output: distance_loop() and distance_number
1)  Suppose club is in a) Putting status; or b) Chipping status; or c) Full swing
    status. Then Let R = MAX_LOOP_STEP_PUTT,
    MAX_LOOP_STEP_CHIP, MAX_LOOP_STEP_NORMAL, respectively.
2)  distance_number  = distance / R
3)  For k = 0 To distance_number-1
4)    distance_loop(k) = R
5)  Next k
6)  If (distance_number >= 1 ) Then
7)    distance_number = distance_number – 1
8)  Else
9)    distance_loop(distance_number) = distance
10) End If
```

Fig. 11